Arduino and Python: Learn Serial Programming

Written By: Chandler

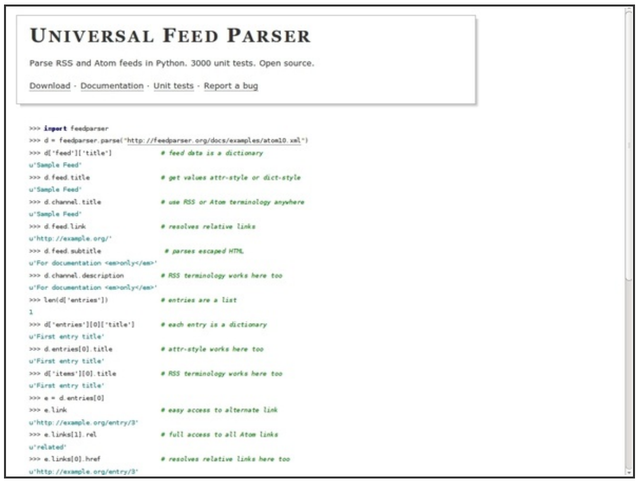
PARTS:

- Arduino microcontroller (1)
- LED (1)
- 16x2 LCD (1)
- Speaker (1)
- 22 awg wire (1)
- Speaker wire (1)

SUMMARY

For this tutorial, you will need (or at least it is helpful to have) a knowledge of the Python programming language. You will also need an Arduino and, depending on how many of the projects below you want to work on, different electronic parts. Let's dive in!

Step 1 — Arduino and Python: Learn Serial Programming

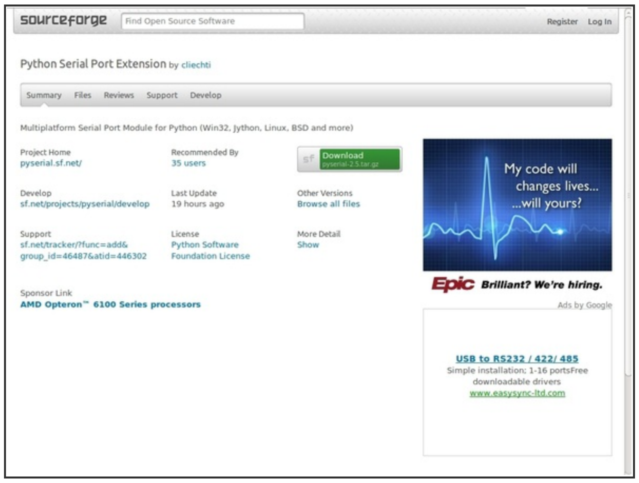


Universal Feed Parser

Parse RSS and Atom feeds in Python. 3000 unit tests. Open source.

Download · Documentation · Unit tests · Report a bug

```
>>> import feedparser
>>> d = feedparser.parse("http://feedparser.org/docs/examples/atom10.xml")
>>> d['feed']['title']          # feed data is a dictionary
u'Sample Feed'
>>> d.feed.title                # get values attr-style or dict-style
u'Sample Feed'
>>> d.channel.title             # use RSS or Atom terminology anywhere
u'Sample Feed'
>>> d.feed.link                 # resolves relative links
u'http://example.org/'
>>> d.feed.subtitle              # parses escaped HTML
u'For documentation <em>only</em>'
>>> d.channel.description       # RSS terminology works here too
u'For documentation <em>only</em>'
>>> len(d['entries'])           # entries are a list
1
>>> d['entries'][0]['title']    # each entry is a dictionary
u'First entry title'
>>> d.entries[0].title          # attr-style works here too
u'First entry title'
>>> d['items'][0].title         # RSS terminology works here too
u'First entry title'
>>> e = d.entries[0]
>>> e.link                      # easy access to alternate link
u'http://example.org/entry/3'
>>> e.links[1].rel              # full access to all Atom links
u'related'
>>> e.links[0].href             # resolves relative links here too
u'http://example.org/entry/3'
```


sourceforge
Find Open Source Software
Register
Log In
Summary
Files
Reviews
Support
Develop
Download
pyserial-2.5.tar.gz
My code will changes lives... ...will yours?
Epic Brilliant? We're hiring.
Ads by Google

Python Serial Port Extension by cliechti

Multiplatform Serial Port Module for Python (Win32, Jython, Linux, BSD and more)

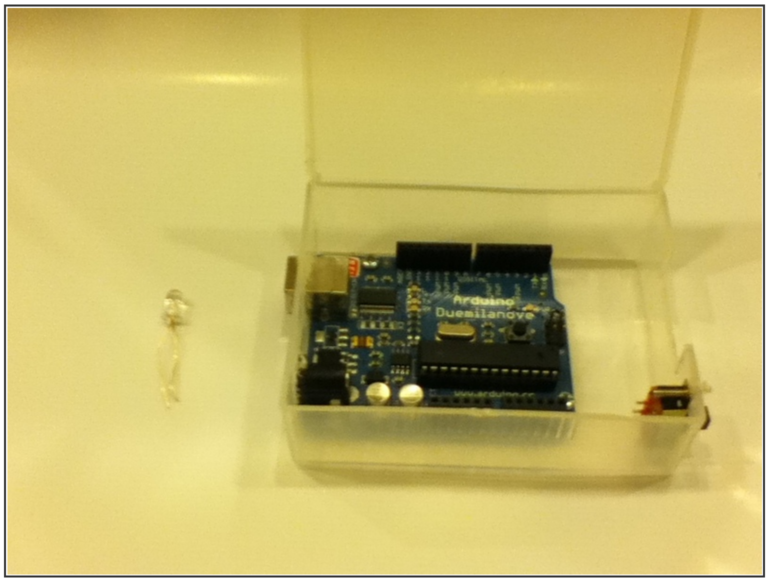
Project Home pyserial.sf.net/	Recommended By 35 users	
Develop sf.net/projects/pyserial/develop	Last Update 19 hours ago	Other Versions Browse all files
Support sf.net/tracker/?func=add&group_id=46487&atid=446302	License Python Software Foundation License	More Detail Show

Sponsor Link
AMD Opteron™ 6100 Series processors

USB to RS232 / 422/ 485
Simple installation; 1-16 portsFree downloadable drivers
www.easysync-ltd.com

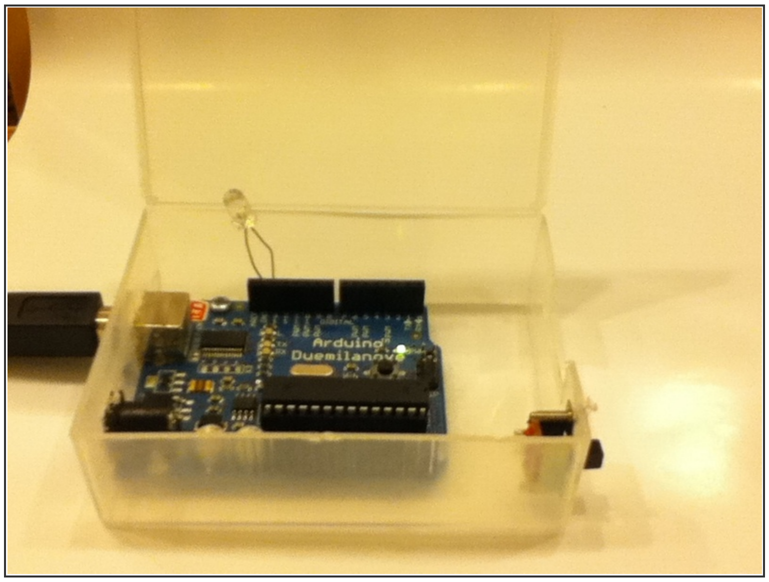
- To start, let's grab your (virtual) parts. Go to:
- http://www.feedparser.org/
- and http://sourceforge.net/projects/pyserial...
- so you can download the packages for the two extra libraries we will need, feedparser and pySerial.
- Note: To install these, unpack the tar.gz file and run setup.py like the following (on Linux):
- python setup.py install
- On other systems, just run the setup.py file with the command line argument "install."

Step 2



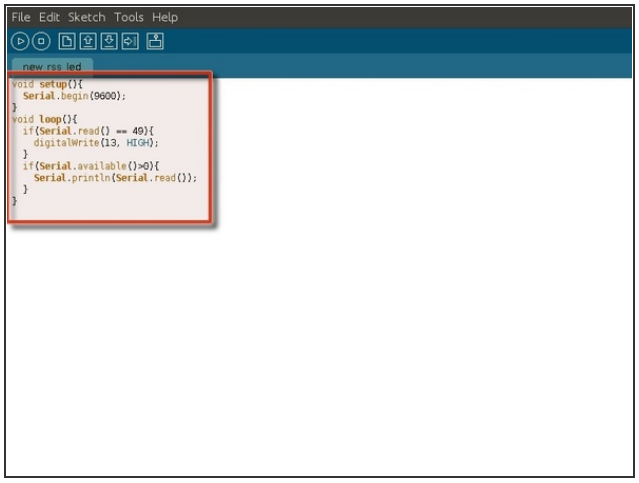
- Now for the physical parts. As you can see to the left, all you need for this first project are an LED and an Arduino. The box and switch are part of my Arduino housing and aren't required.

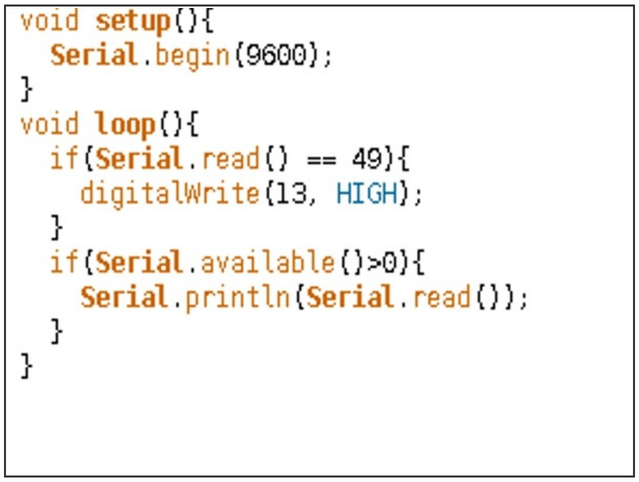
Step 3



- Now just connect the LED between pin 13 (+) and GND (-), and plug the Arduino into your computer.

Step 4

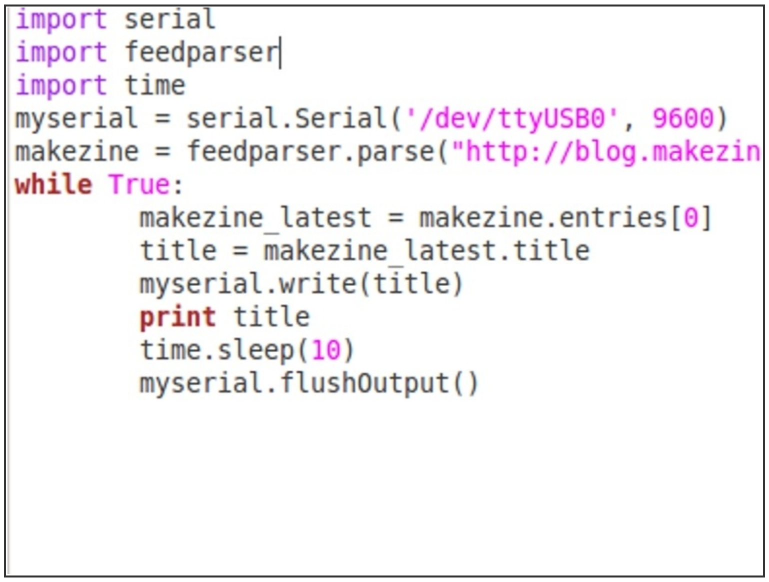




```
void setup(){
  Serial.begin(9600);
}
void loop(){
  if(Serial.read() == 49){
    digitalWrite(13, HIGH);
  }
  if(Serial.available()>0){
    Serial.println(Serial.read());
  }
}
```

- Now, just download this program onto the Arduino. Leave it plugged in after downloading.

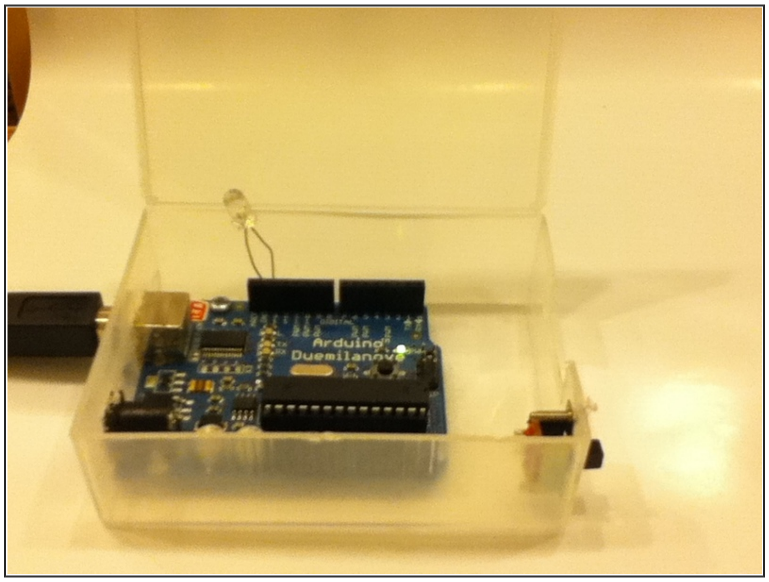
Step 5



```
import serial
import feedparser
import time
myserial = serial.Serial('/dev/ttyUSB0', 9600)
makezine = feedparser.parse("http://blog.makezin
while True:
        makezine_latest = makezine.entries[0]
        title = makezine_latest.title
        myserial.write(title)
        print title
        time.sleep(10)
        myserial.flushOutput()
```

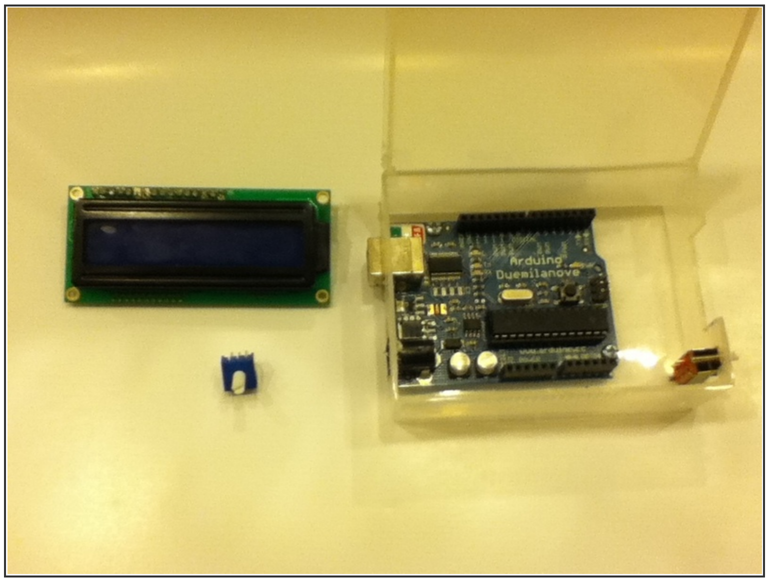
- Then, create a file called `new_rss.py`, or whatever you want, ending in ".py". However, make sure to take note of what you call it. Inside the file, write the text in the accompanying image.
- Note: Sorry, but the image got cut off after cropping. The cut-off line says:
- makezine = feedparser.parse("http://blog.makezine.com/feed")

Step 6



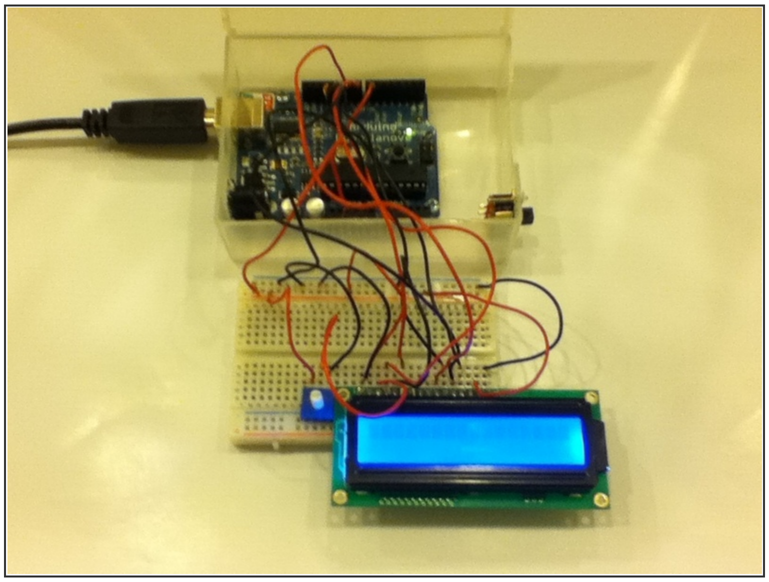
- Now, make sure that your Arduino is still connected and running the Arduino sketch you made earlier. Now, run the Python program, and when a new feed comes in from the Make website, the LED should light up!
- Note: To view other websites, in the line that was cut off in the earlier image change the URL to the one for the desired website. The URL must be that of an Atom/RSS feed.

Step 7



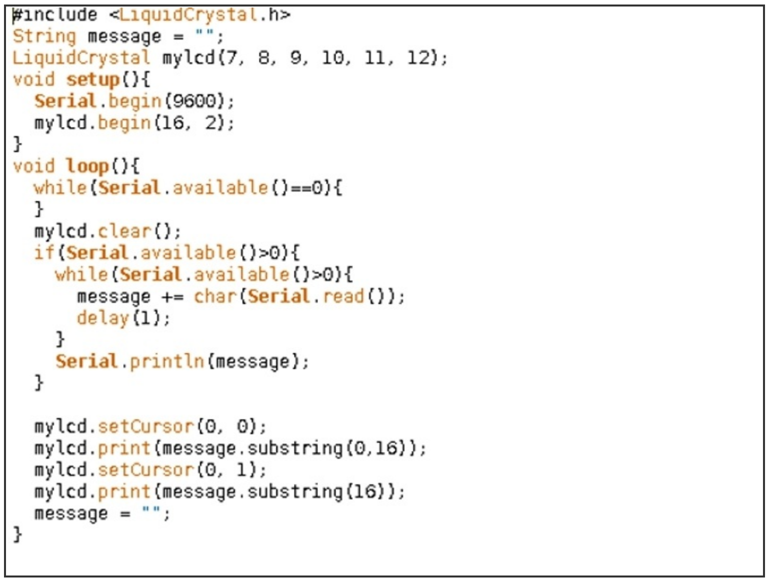
- Great! On to project #2! Our next project goal is to print the title of the latest RSS feed entry onto an LCD screen. First, assemble the parts, plus breadboard and wires.
- Note: The LCD screen is from the Maker Shed.

Step 8



- Then, wire up the LCD screen according to the instructions on the Maker Shed site.

Step 9

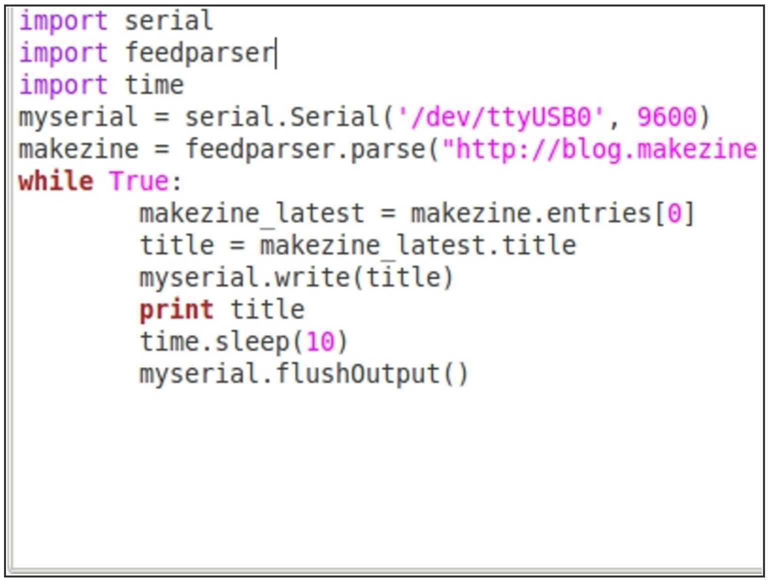


```
#include <LiquidCrystal.h>
String message = "";
LiquidCrystal mylcd(7, 8, 9, 10, 11, 12);
void setup(){
  Serial.begin(9600);
  mylcd.begin(16, 2);
}
void loop(){
  while(Serial.available()==0){
  }
  mylcd.clear();
  if(Serial.available()>0){
    while(Serial.available()>0){
      message += char(Serial.read());
      delay(1);
    }
    Serial.println(message);
  }

  mylcd.setCursor(0, 0);
  mylcd.print(message.substring(0,16));
  mylcd.setCursor(0, 1);
  mylcd.print(message.substring(16));
  message = "";
}
```

- Let's program it! Go ahead and type up the code in the box to the left and download it to the Arduino. Sorry about the size of the text.

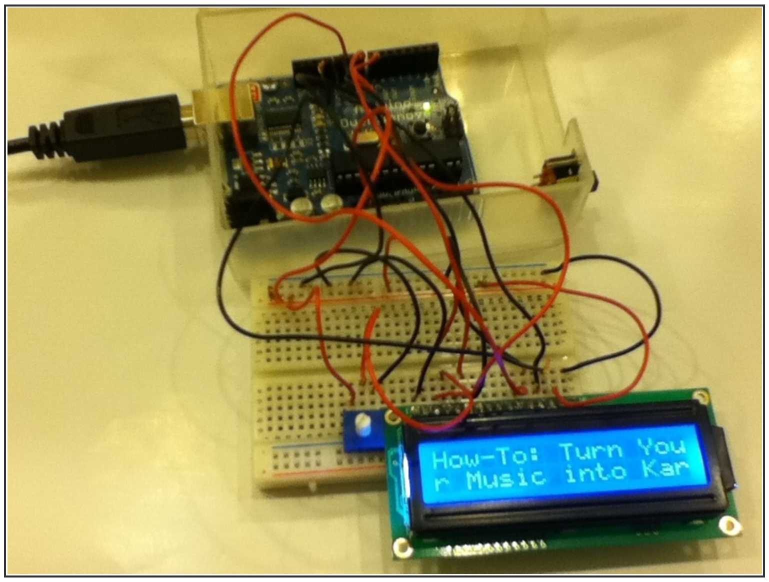
Step 10



```
import serial
import feedparser
import time
myserial = serial.Serial('/dev/ttyUSB0', 9600)
makezine = feedparser.parse("http://blog.makezine
while True:
        makezine_latest = makezine.entries[0]
        title = makezine_latest.title
        myserial.write(title)
        print title
        time.sleep(10)
        myserial.flushOutput()
```

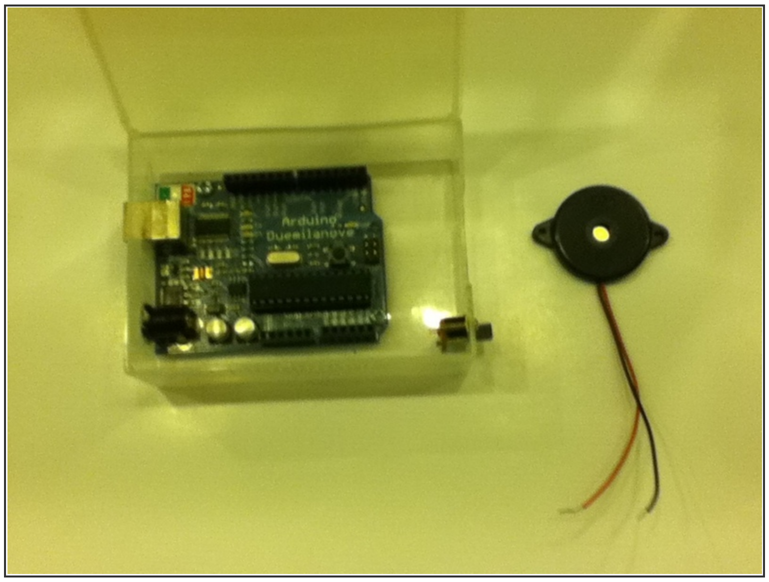
- Now let's write the Python script. This one can also be named anything, but for a suggestion, something along the lines of `make_rss_read.py` should do.
- Note: Once again, the last line accidentally got clipped. Oddly enough, the same line got clipped. The complete line reads:
- makezine = feedparser.parse("http://blog.makezine.com/feed")

Step 11



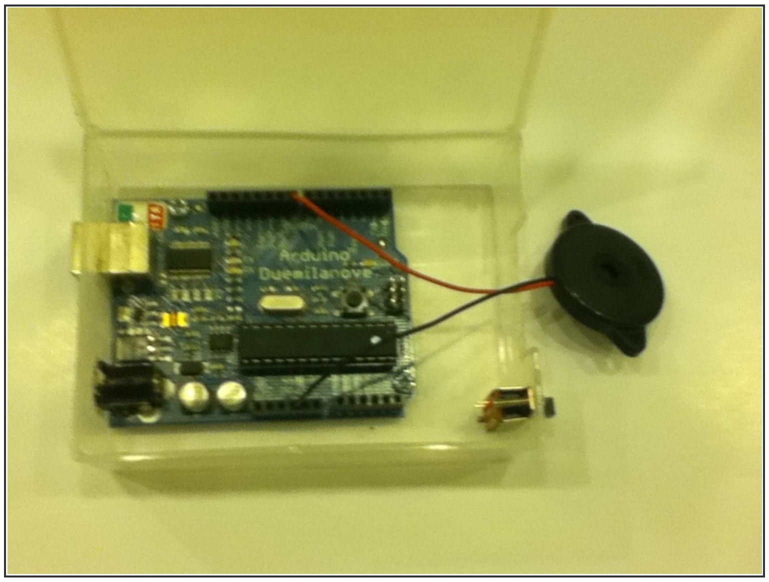
- Run this code, and the latest RSS feed from Make should display!

Step 12



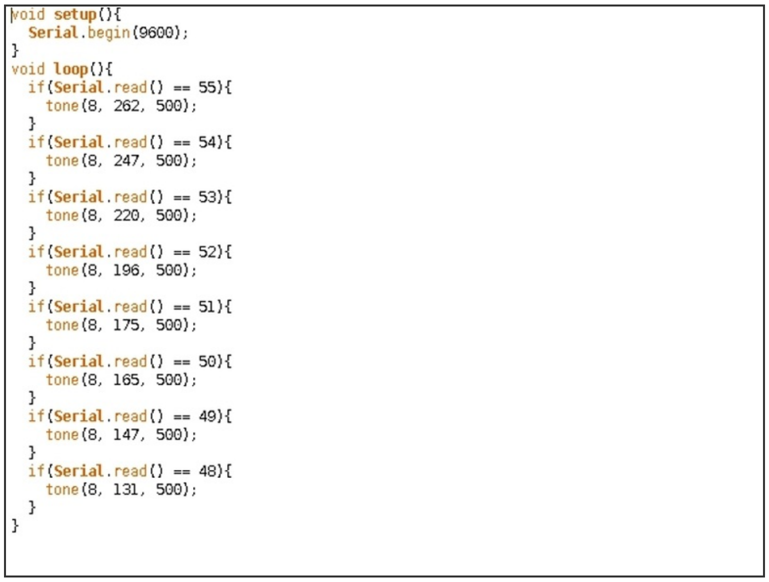
- Last but not least, number 3! Our final goal is to create a virtual piano. There will be a GUI (graphical user interface) where the user clicks buttons to play notes.
- To begin this one, we have some physical parts to collect: an Arduino and a speaker.

Step 13



- To assemble this project, simply plug the speaker into the Arduino, with the positive pin going to pin 8 and the negative to ground. Also, plug in the Arduino to your computer.

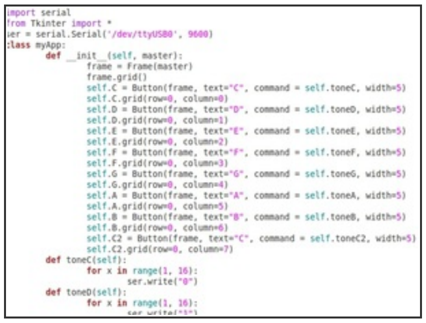
Step 14



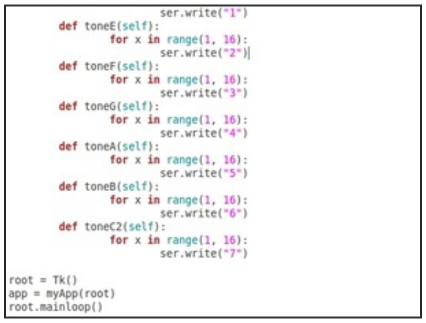
```
void setup(){
  Serial.begin(9600);
}
void loop(){
  if(Serial.read() == 55){
    tone(8, 262, 500);
  }
  if(Serial.read() == 54){
    tone(8, 247, 500);
  }
  if(Serial.read() == 53){
    tone(8, 220, 500);
  }
  if(Serial.read() == 52){
    tone(8, 196, 500);
  }
  if(Serial.read() == 51){
    tone(8, 175, 500);
  }
  if(Serial.read() == 50){
    tone(8, 165, 500);
  }
  if(Serial.read() == 49){
    tone(8, 147, 500);
  }
  if(Serial.read() == 48){
    tone(8, 131, 500);
  }
}
```

- Now for the Arduino code. Download this program onto your Arduino, and leave it plugged in. Once again, sorry for the small text.

Step 15

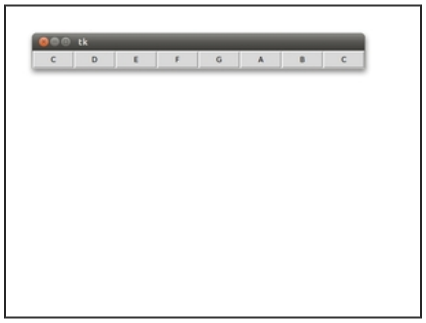


```
import serial
from Tkinter import *
ser = serial.Serial('/dev/ttyUSB0', 9600)
class myApp:
        def __init__(self, master):
                frame = Frame(master)
                frame.grid()
                self.C = Button(frame, text="C", command = self.toneC, width=5)
                self.C.grid(row=0, column=0)
                self.D = Button(frame, text="D", command = self.toneD, width=5)
                self.D.grid(row=0, column=1)
                self.E = Button(frame, text="E", command = self.toneE, width=5)
                self.E.grid(row=0, column=2)
                self.F = Button(frame, text="F", command = self.toneF, width=5)
                self.F.grid(row=0, column=3)
                self.G = Button(frame, text="G", command = self.toneG, width=5)
                self.G.grid(row=0, column=4)
                self.A = Button(frame, text="A", command = self.toneA, width=5)
                self.A.grid(row=0, column=5)
                self.B = Button(frame, text="B", command = self.toneB, width=5)
                self.B.grid(row=0, column=6)
                self.C2 = Button(frame, text="C", command = self.toneC2, width=5)
                self.C2.grid(row=0, column=7)
        def toneC(self):
                for x in range(1, 16):
                        ser.write("0")
        def toneD(self):
                for x in range(1, 16):
                        ser.write("1")
```



```
                        ser.write("1")
        def toneE(self):
                for x in range(1, 16):
                        ser.write("2")
        def toneF(self):
                for x in range(1, 16):
                        ser.write("3")
        def toneG(self):
                for x in range(1, 16):
                        ser.write("4")
        def toneA(self):
                for x in range(1, 16):
                        ser.write("5")
        def toneB(self):
                for x in range(1, 16):
                        ser.write("6")
        def toneC2(self):
                for x in range(1, 16):
                        ser.write("7")

root = Tk()
app = myApp(root)
root.mainloop()
```



- Now, save the text to the left as `digital_piano.py`, or whatever you want, ending in ".py". (Note that this code was adapted from the tutorial: *An Introduction to Tkinter*. Thanks!) Also, the code is shown in two images. The images, however, have some overlaps, so watch out for that. Finally, the first letters of the first three lines are i, f, and s, in case it was too hard to tell.
- When you're done with that, execute the file. A window should pop up! To use this, click a button and the note will play from the Arduino!
- Also, an image of the final product is shown to the left.

Step 16

- Additional notes: If you had any trouble with these projects, go into the Python file and make sure that the path to the Arduino (currently /dev/ttyUSB0) is the path to yours. The path above only works for Ubuntu, and so you may have to change it around. Also, the piano is in its beta version, and has a few glitches.
- Final note: If these projects interested you, look at the pySerial documentation that can easily be found via Google.
- Please go ahead and modify/tinker with these projects, and create your own. I have already created an amazing TVout application that prints off RSS feeds, and did it in about one day. Use this tutorial, and post your accomplishments here, at Make: Projects.

This document was last generated on 2012-10-31 10:32:41 AM.